SMART: Analyzing the Reuse Potential of Legacy Systems in Service- Oriented Architecture (SOA) Environments

Grace A. Lewis
Research, Technology and Systems Solutions (RTSS) Program
System of Systems Practice (SoSP) Initiative

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **09 APR 2009**	2. REPORT TYPE	3. DATES COVERED **00-00-2009 to 00-00-2009**

4. TITLE AND SUBTITLE **SMART: Analyzing the Reuse Potential of Legacy Systems in Service-Oriented Architecture (SOA) Environments**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,Software Engineering Institute,Pittsburgh,PA,15213**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**40**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Speaker Biography

Grace Lewis is a Senior Member of the Technical Staff at the Software Engineering Institute. She is currently the lead for the System of Systems Engineering team within the System of Systems Practice (SoSP) initiative. Her current interests and projects are in service-oriented architecture, technologies for systems interoperability, modernization of legacy systems, and characterization of software development life cycle activities in systems of systems environments. Her latest publications include several reports published by Carnegie Mellon on these subjects and a book in the SEI Software Engineering Series. She is also a member of the technical faculty for the Master in Software Engineering program at CMU. Grace holds a B.Sc. in Systems Engineering and an Executive MBA from Icesi University in Cali, Colombia; and a Master in Software Engineering from Carnegie Mellon University.

Agenda

SOA Basics ⇦

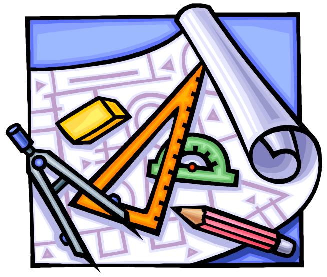
SMART (Service Migration and Reuse Technique)

Summary

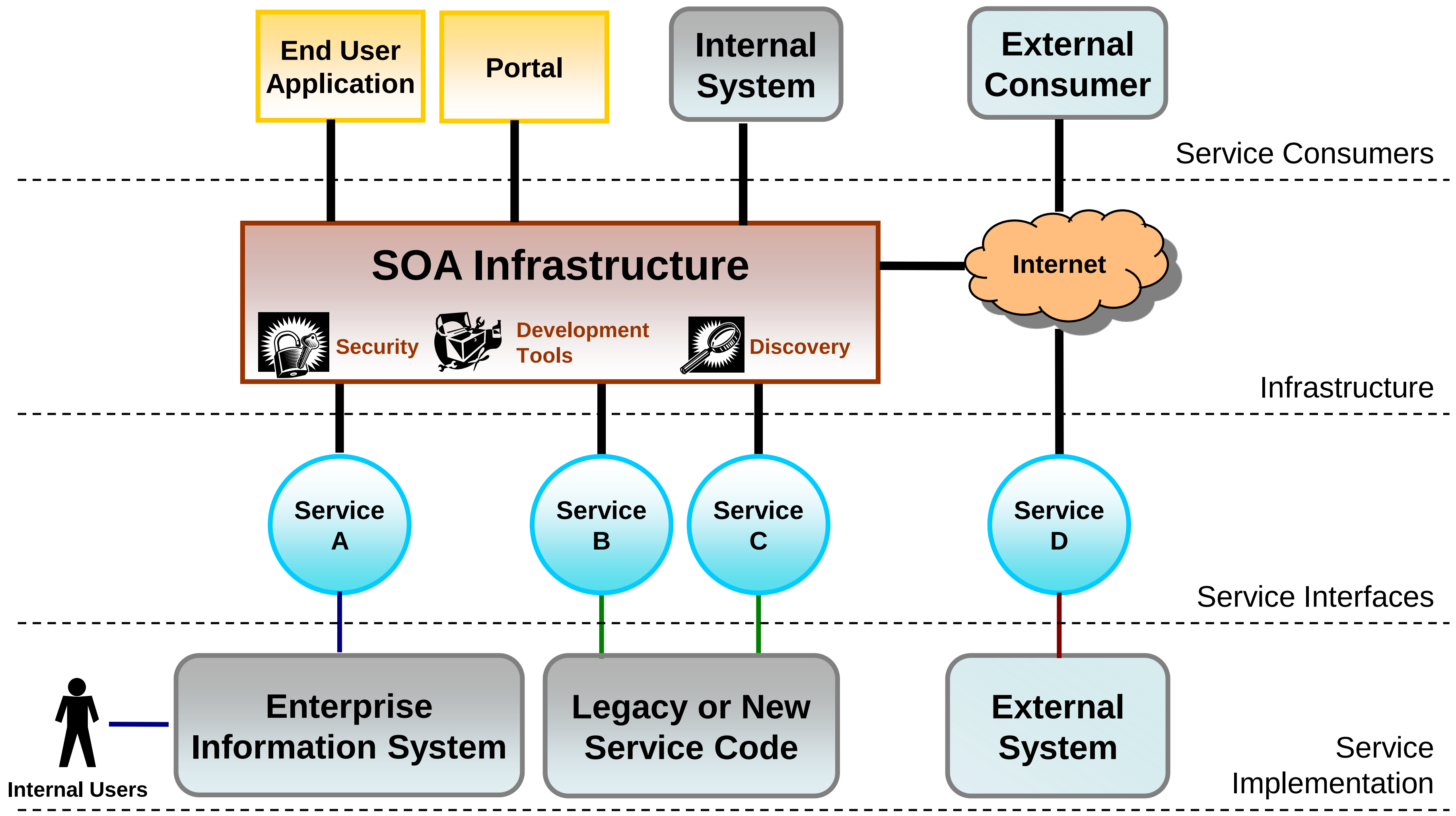
What is SOA?

Service-oriented architecture is a way of designing, developing, deploying and managing systems, in which

- Services provide reusable business functionality with well-defined interfaces.
- Service consumers are built using functionality from available services.
- Service interface definitions are first-class artifacts.
  - There is clear separation of service interface from service implementation.
- An SOA infrastructure enables discovery, composition, and invocation of services.
- Protocols are predominantly, but not exclusively, message-based document exchanges.



Components of a Service-Oriented System



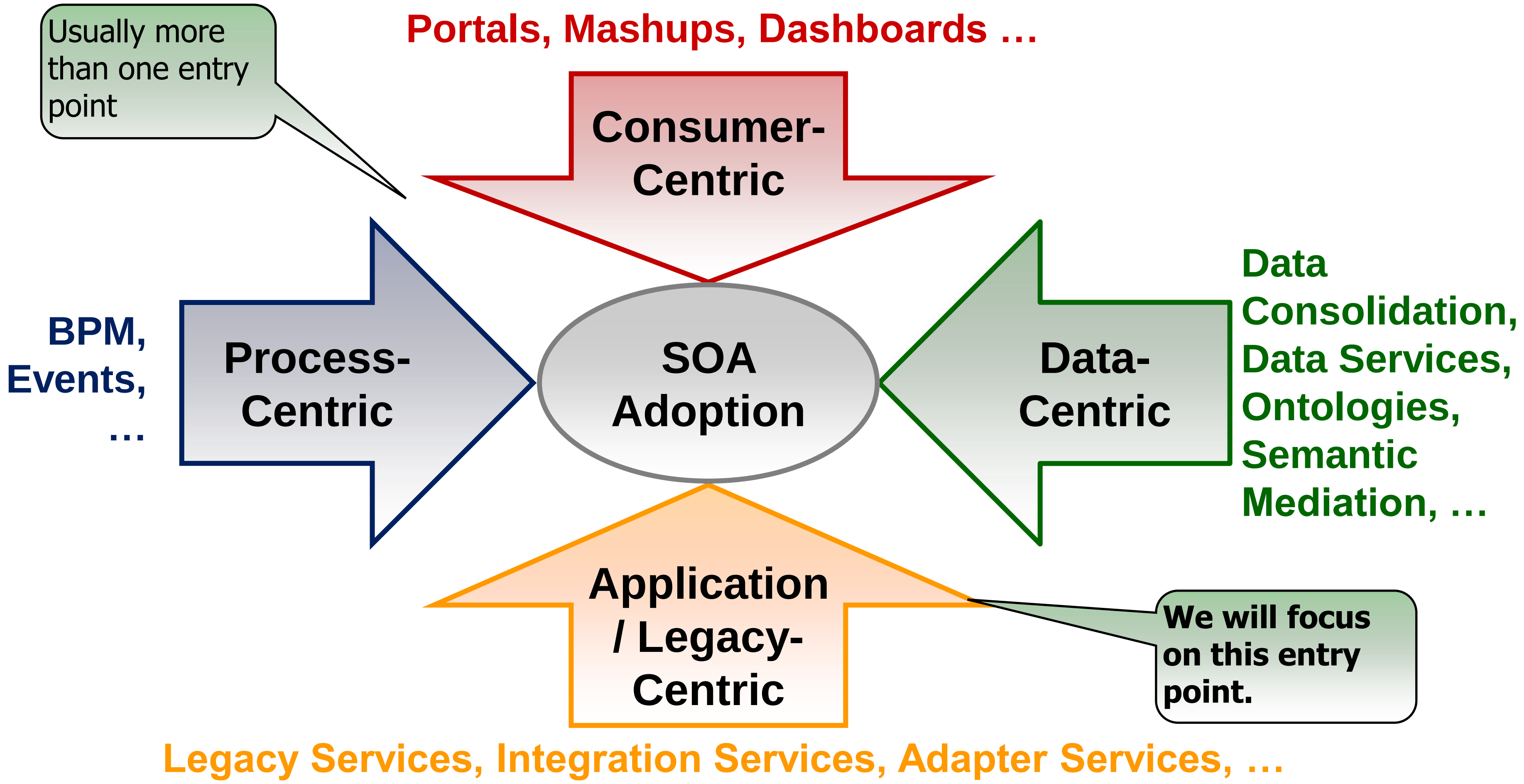
Agenda

SOA Basics

SMART (Service Migration and Reuse Technique) ⇦

Summary

SOA Entry Points


Usually more than one entry point
Portals, Mashups, Dashboards …
Consumer-Centric
BPM, Events, …
Process-Centric
SOA Adoption
Data-Centric
Data Consolidation, Data Services, Ontologies, Semantic Mediation, …
Application / Legacy-Centric
We will focus on this entry point.
Legacy Services, Integration Services, Adapter Services, …

Source: Adapted from AgilePath's SOA Quad Model™

Legacy System Reuse in the SOA Context

Reuse at a higher level

- Reuse of business functionality
- Encapsulation of technical details

Reuse across organizations

- Organizations can "sell" their core business expertise as services.
- Functionality can be acquired as opposed to developed from scratch—potential savings.

Option for leveraging legacy system investment

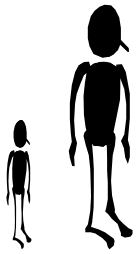
- In many cases, legacy components can be reused by exposing them as services, independent of vendor, platform, and technology.

Legacy System Reuse Challenges

Reuse at the service level is more complex than reuse at the module or component level.

- From the service provider perspective
  - Designing reusable services requires a different approach, skill set, and mindset
  - Bigger stakeholder community because services are typically reused at organization and sub-organization level
  - Services need to be as generic as possible so that they are of interest to multiple service consumers and at the same time need to add value to potential consumers
- From the service consumer perspective
  - Larger granularity may lead to larger incompatibilities

Challenges can come from the legacy system from itself or from the environment.



Bottom Line

There are issues to take into consideration that go beyond adding a service interface to an existing system.

SMART is an approach to make initial decisions about the feasibility of reusing legacy systems within an SOA environment.

SMART: Service Migration and Reuse Technique

The end goal for SMART is the identification a pilot project that will help shape a migration strategy for an organization, along with an understanding of cost and risk involved.

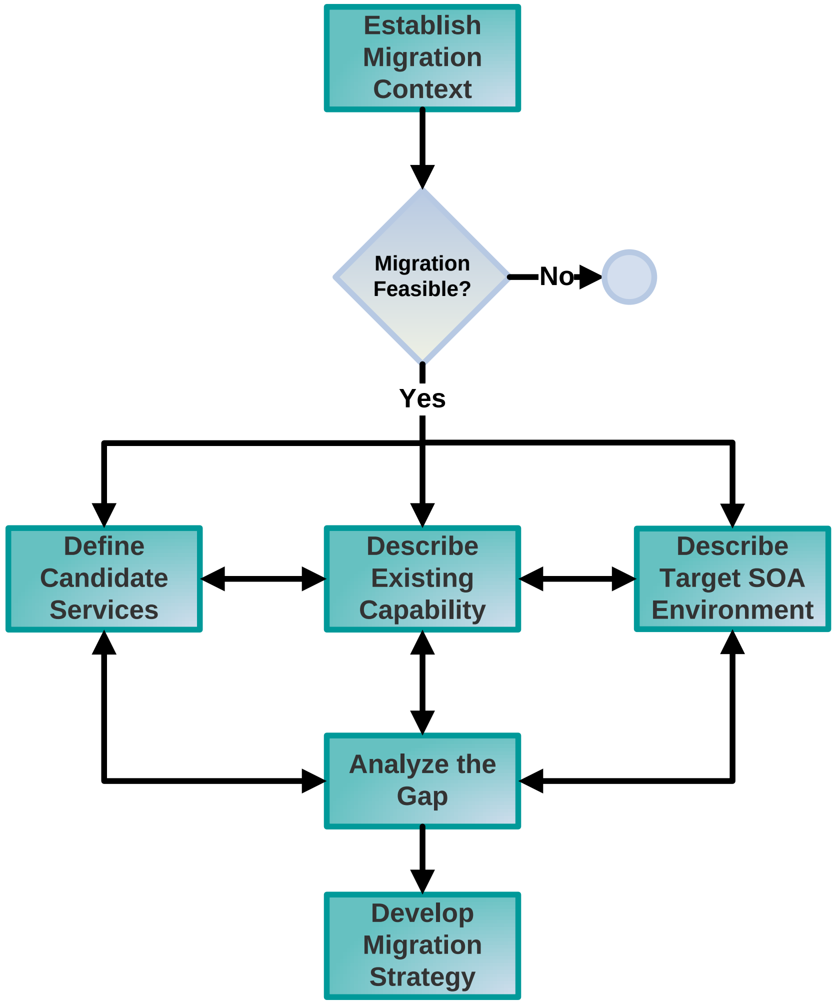
SMART analyzes the viability of reusing legacy systems in an SOA environment:

- Does it make sense to migrate the legacy system to services?
- What services make sense to develop?
- What legacy system components can be used to implement these services?
- What changes to components are needed to accomplish the migration?
- What migration strategies are most appropriate?
- What are the preliminary estimates of cost and risk?
- What is an ideal pilot project that can help address some of these risks?

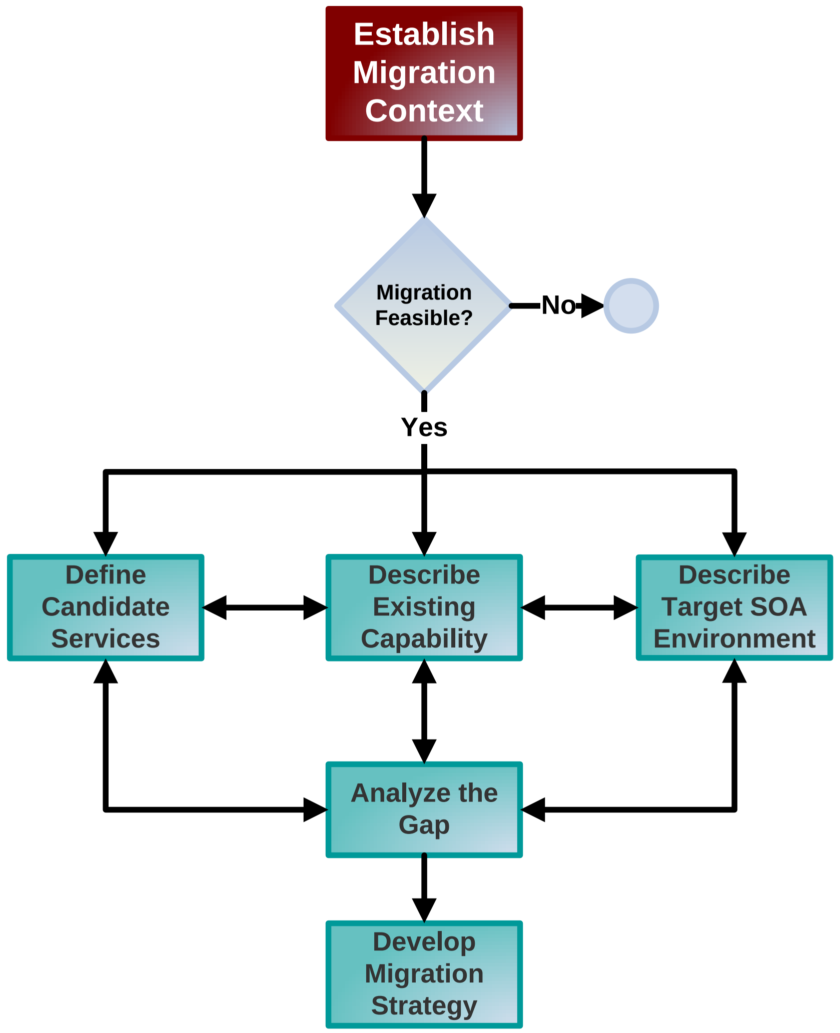
Three Elements of SMART

Process 1	SMART Interview Guide (SMIG) 2	Artifacts 3
Gathers information about • Goals and expectations of migration effort • Candidate services • Legacy systems • Target SOA environment Analyzes gap between legacy and target state	Guides discussions in initial SMART activities	• Stakeholder List • Characteristics List • Migration Issues List • Business Process-Service Mapping • Service Table • Component Table • Notional Service-Oriented System Architecture • Service-Component Alternatives • Migration Strategy

SMART Process Activities


Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

Establish Migration Context


Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

Understand the business and technical context for migration

- Rationale, goals and expectations
- Technical and business drivers
- Programmatic constraints (e.g. schedule, budget)
- Previous related efforts or analyses

Identify stakeholders

- Who is driving and paying for the effort
- Who knows what about the legacy system and the target SOA environment
- Demand or need for potential services

Understand legacy system and target SOA environment at a high level

Identify a set of candidate services for migration

Establish Migration Context: SMIG Examples

Discussion Topic	Related Questions	Potential Migration Issues
Goal and Expectations of Migration Effort	• What are the business and technical drivers for the migration effort? • What are the short-term and long-term goals?	• No SOA strategy • Goals for migration are not clear.
High-Level Understanding of Legacy System	• What is the main functionality provided by the legacy system? • What is the high-level architecture of the system? • What is the current user interface to the system?	• Legacy system knowledge is not available. • Architectural mismatch • User interface complexity hard to replicate in service consumers
High-Level Understanding of Target SOA Environment	• What are the main components in the target SOA environment? • Is this the organization’s first attempt to deploy services in this environment?	• Target SOA environment has not been identified. • No in-house knowledge of target SOA environment
Potential Service Consumers	• Who are the potential service consumers?	• Consumers for services have not been identified.

Case Study: Establish Migration Context 1

DoD organization tasked with developing services that can be used by mission planning and execution applications

MSS is a system for comparison of planned mission against current state to determine if corrective actions should be taken

- In final stages of development

Drivers

- Migration to services was already a longer-term goal for MSS
- Make developed services available to all mission planning and execution systems

Requirement to demonstrate the feasibility of one component as a service being used by one mission planning and execution system within 6 months and to migrate the full system to services in two years

Case Study: Establish Migration Context $_2$

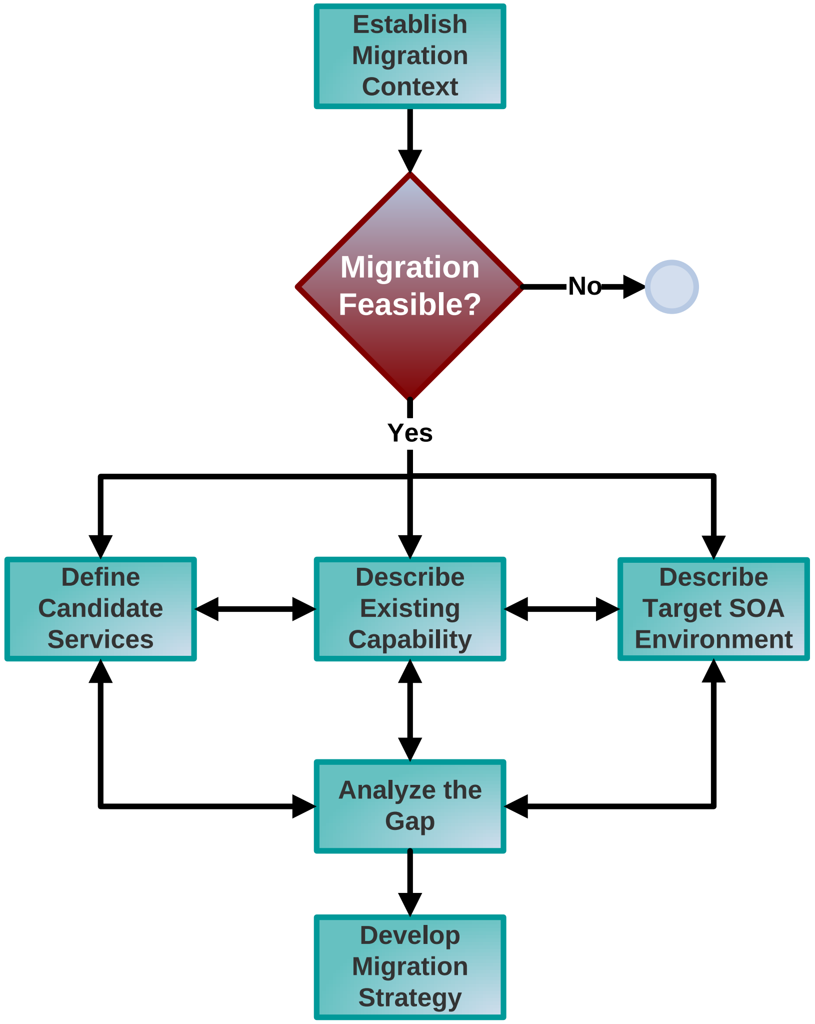
Standard Web Services environment is target SOA environment

- Not clear that this will be the future environment for the developed services

Representatives from the legacy system and a representative from a mission planning and execution application (service consumer) agreed on the following candidate services

- *AvailablePlans*: Provides list of available plans that are being reasoned about.
- *TrackedTasksPerPlan*: Provides list of tasks that are being tracked for a certain plan.
- *TaskStatus*: Provides the status for a given task in a given plan.
- *SetTaskAlert*: Alerts when a given task in a given plan satisfies a certain condition

Checkpoint for Migration Feasibility



Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

Decision to continue with the process has to be made

Potential outcomes at this point are

- The migration is initially feasible
- The migration has potential but requires additional information to make an informed decision
- The migration is not feasible

Case Study: Migration Feasibility

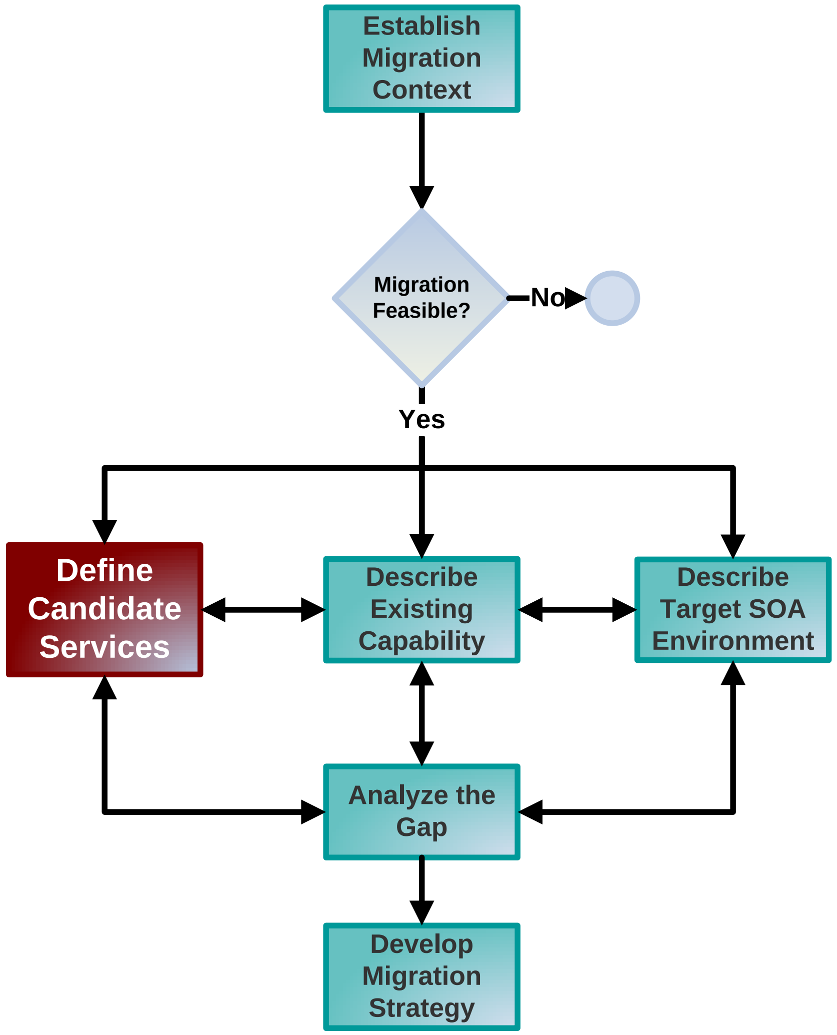
Decision: Migration feasible

- Availability of stakeholders from the service provider and a service consumer
- Good understanding of the legacy system
- Request-response nature of the identified services
- Reasonable initial mapping of services to components

Migration issues identified in this activity

- Short-term goal for the migration is different from long-term goal migration
  - Work to accomplish the short-term goal might have to be redone in order to accomplish the long-term goal
- System is a single-user, single-plan system
  - When capabilities are migrated to services, it will have to support multiple users and multiple plans

Define Candidate Services



Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

Select a small number of services, usually 3-4, from the initial list of candidate services

For these candidate services, the end goal is to fully specify inputs and outputs

Case Study: Define Candidate Services

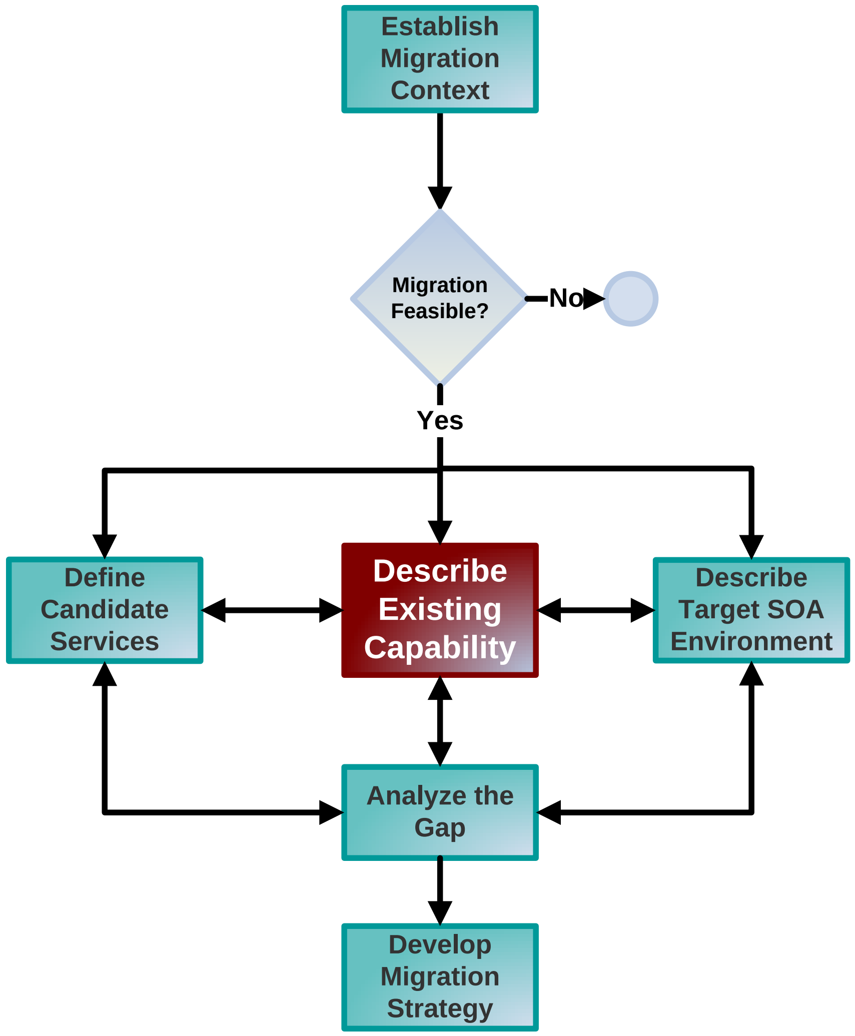
The list of services identified in the previous step was considered reasonable for analysis

Inputs and outputs were next identified in detail for each of these services

Migration issues identified in this activity

- *SetTaskAlert* requires (1) alert is set up to respond to certain conditions and (2) service consumer is alerted when the condition is reached
  - Handling of events in service-oriented environments is relatively new—SOA 2.0
- Unclear how the alert mechanism is going to be implemented
  - SOA infrastructure would need to have a way to call back the service consumer
  - There might also be firewall issues on the consumer side
- Complexity of alert conditions is high
  - Service consumer interface will have to replicate this complexity or conditions would have to be made simpler or limited

Describe Existing Capability


Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

Obtain descriptive data about legacy components

- Name, function, size, language, operating platform, age of legacy components, etc.

Question technical personnel about

- Architecture and design paradigms
- Complexity, coupling, interfaces
- Quality of documentation
- Component/product dependencies

Gather data about

- Quality, maturity, existing problems
- Change history
- User satisfaction

Describe Existing Capability: SMIG Examples

Discussion Topic	Related Questions	Potential Migration Issues
Legacy System Characteristics	• What is the history of the system? • Is the system a proof of concept, prototype, under development, in testing, or a fielded system? • What system documentation is available? • Does the system have interfaces to other systems? • What are potential locking, persistence, or transaction problems if accessed by multiple users when migrated to services?	• Planned development concurrent with service migration • Limited system documentation • Interfaces to other systems will open doors to service consumers. • Single-user system may have problems in a multi-user environment.
Legacy System Architecture	• What architecture views are available? • What are the major modules of the system and dependencies between modules? • Is user interface code separate from the business logic code? • Are there any design paradigms or patterns implemented in the system? • What are the key quality attributes built into the current architecture of the system?	• Lack of architecture documentation may lead to underestimation of complexity. • Tight coupling between user interface code and business logic code increases effort. • Undocumented violations of design patterns may cause problems. • Key quality attributes may not hold true in a services environment.
Code Characteristics	• What code documentation is available? • What coding standards are followed?	• Poor coding practices will increase migration effort.

Case Study: Describe Existing Capability

MSS characteristics

- In demonstration state
- Written in C++, C# and Managed C++ in a Visual Studio 2005 development environment
- Runs on a Windows XP platform
- Size of the full system is approximately 13,000 lines of code
- Code documentation was rated between *Fair* and *Good* by its developers

Several architecture views were presented that were useful for understanding the system

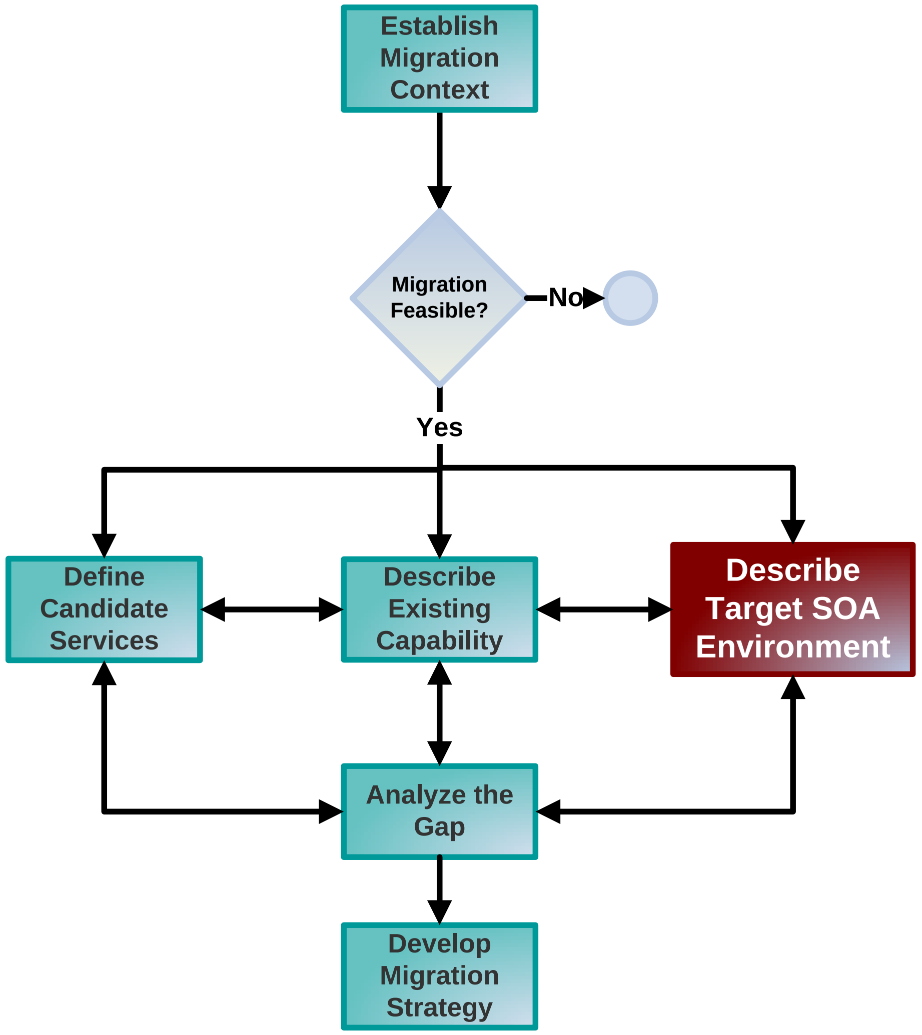
MSS relies on an external planning system (PS) for plan data and situational awareness data

- PS is being targeted for migration to services in the future

Migration issues identified in this activity

- Documentation for most of the analyzed classes was determined *Fair*
  - Could be an issue if original developers do not perform the migration
- Currently a large amount of communication between MSS and PS
  - Unclear how performance will be affected when this communication takes place using services (they currently reside on the same machine)
- Task alert functionality is not currently implemented in MSS
  - Still unknowns about the specifics of the implementation

Describe Target SOA Environment



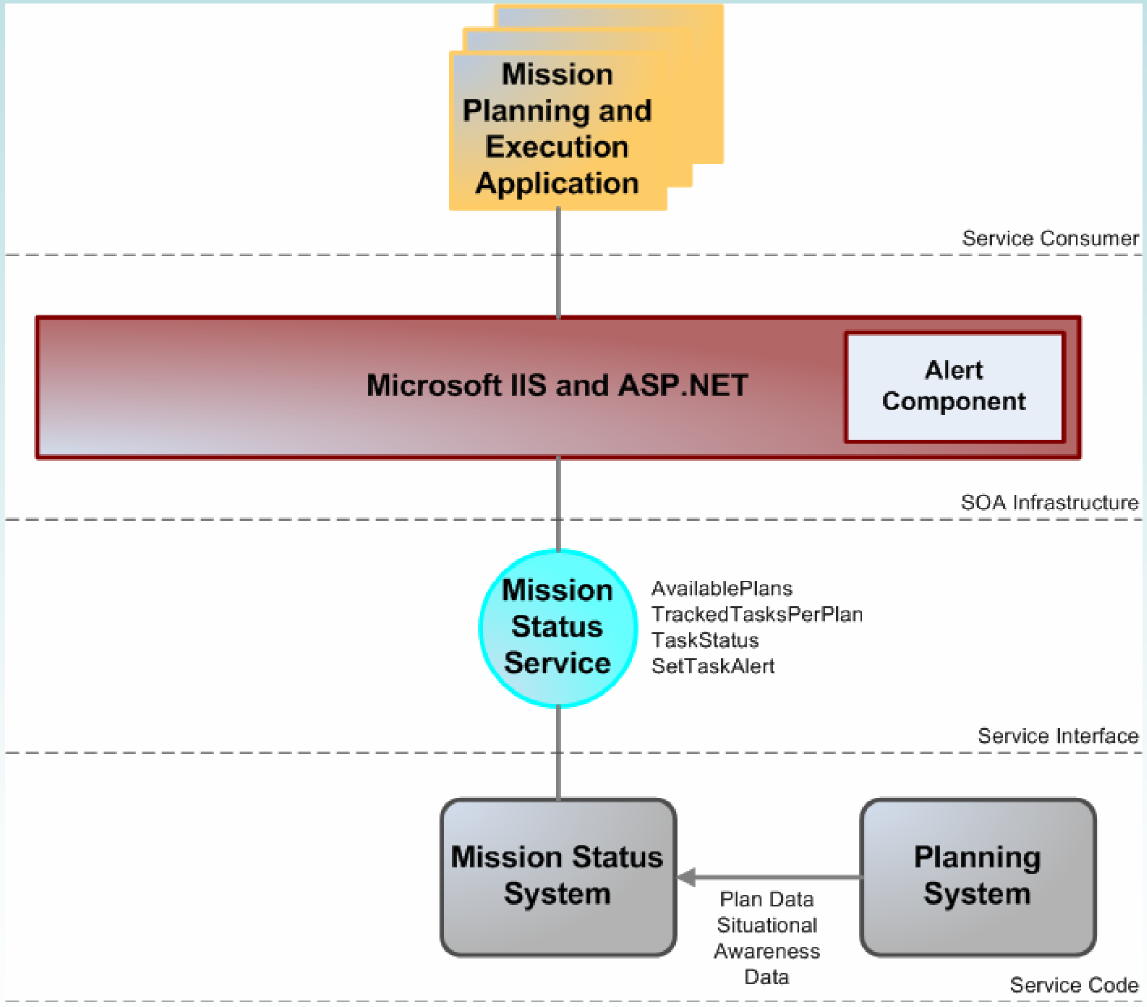
Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

- Identify the impact of specific technologies, standards, and guidelines for service implementation
- Determine state of target SOA environment
- Identify how services would interact with the SOA environment
- Determine QoS expectations and execution environment for services

Describe Target SOA Environment: SMIG Examples

Discussion Topic	Related Questions	Potential Migration Issues
SOA Environment Characteristics	• What is the status of the target SOA environment? • What are the major components of the SOA infrastructure? • Does the target SOA environment provide infrastructure services (i.e., communication, discovery, security, data storage)? • What is the communication model? • What constraints does the target SOA environment impose on services? • Does the legacy system have any behavior that would be incompatible with the target SOA environment? • Once developed, where will services execute?	• Target SOA environment undefined • Redundancy/conflicts between infrastructure services and legacy code • Lack of tools to support legacy code migration to target infrastructure • Compliance with constraints requires major effort. • Architectural mismatch • No thought given to service deployment and execution
Support	• Do you have to provide automated test scripts for the services and make them publicly available? • How will service consumers report problems and provide feedback? • How will service consumers be informed of potential changes in service interfaces and downtime due to upgrades or problems?	• Underestimation of effort to provide service consumer support • Lack of awareness of support requirements

Case Study: Notional Service-Oriented System Architecture

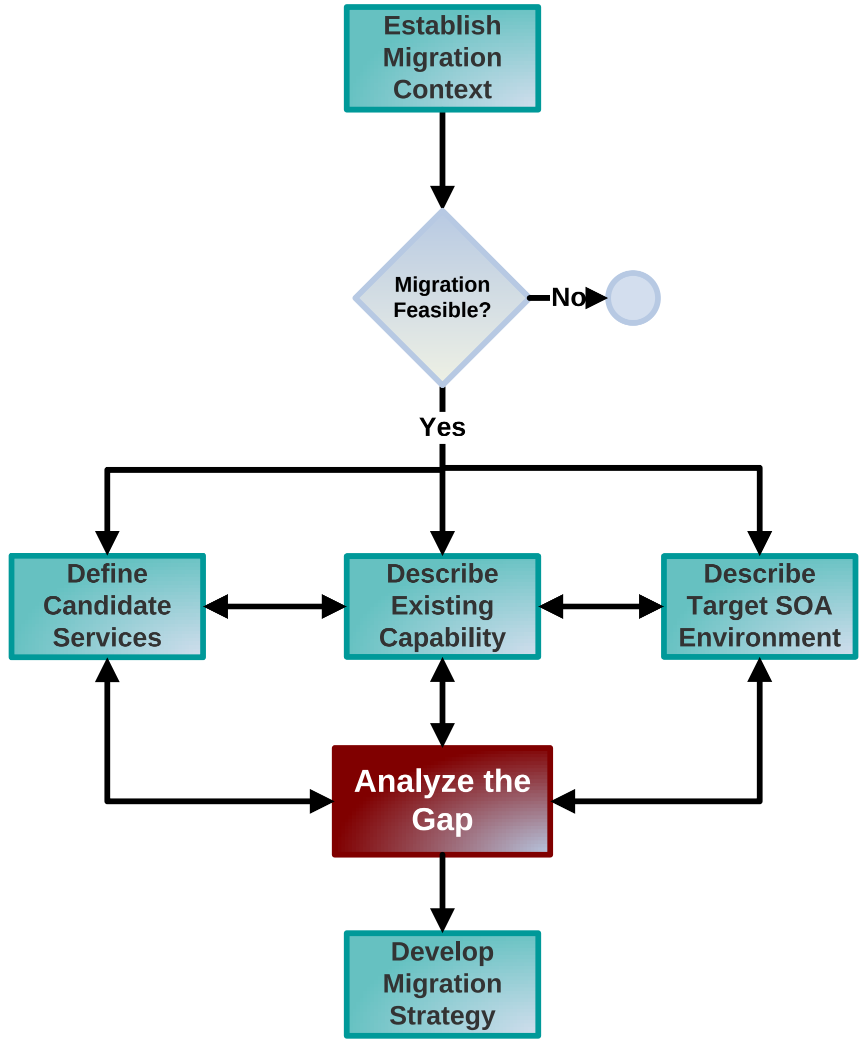


Case Study: Describe Target SOA Environment

Migration issues identified in this activity

- Not known if the identified publish-subscribe component to facilitate alerts will allow someone to subscribe on behalf of a third party
  - If not, the service consumer will have to be aware of the dependency on the publish-subscribe component
  - Ideal situation would be for the *SetTaskAlert* service code to subscribe on behalf of the service consumer, so that the service consumer is not affected if the alert mechanism changes
- If the service consumer has to be set up as a Web server, it would have to be configured so that it accepts incoming messages from the publish-subscribe component
  - Potential security concern

Analyze the Gap


Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

- Define effort, risk and cost to migrate legacy components, given candidate service requirements and target SOA characteristics
- Determine need for additional analyses

Case Study: Analyze the Gap

Developers were asked to

- Describe the details of the changes that would have to be made to the code given the service requirements, the service inputs and outputs, as well as the characteristics and components of the target SOA environment
- Provide an estimate of the effort required to make these changes

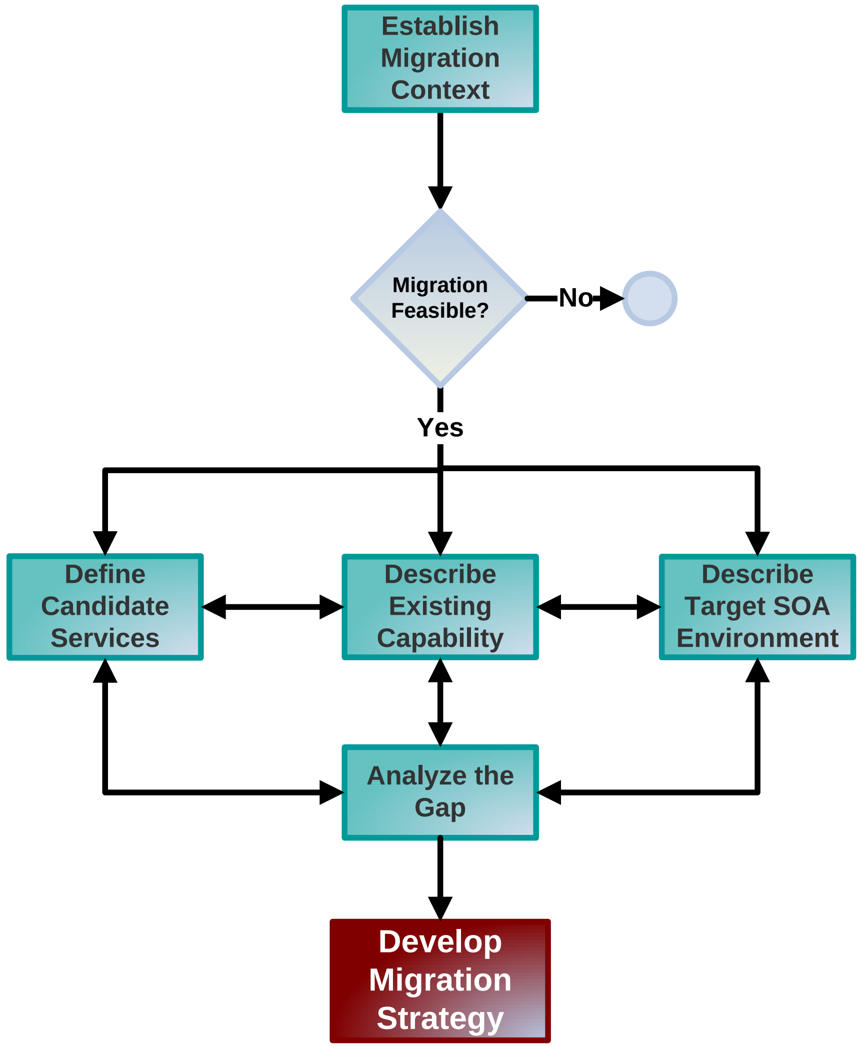
No code analysis or architecture reconstruction was necessary because

- Original developers were involved in the process
- Input was credible
- Architecture documentation and knowledge of the system were acceptable

Exercise: Analyze the Gap—Updated Component Table

COMPONENT		MIGRATION FACTORS		MIGRATION ESTIMATES			
ID	Component Name	Migration Method	Summary of Changes Required	Level of Difficulty	Level of Risk	Effort (Person-weeks)	Cost
1	ComparisonEngine	New + Extraction	1. Add methods to store and retrieve plan name and IDs 2. Add class to process service requests from all 4 services 3. Make changes to handle multiple plans 4. Define structure of a condition	Medium	Low	5	$
2	Analyzer	New + Extraction	1. Add methods to get tasks by plan 2. Modify all methods that retrieve tasks to retrieve tasks per plan	Low	Low	1	$
3	Task	New + Extraction	1. Add methods to get and set plan that a task is connected to 2. Modify constructor to set new attribute 3. Modify toXML and fromXML to serialize and deserialize new attribute	Low	Low	1	$
5	AlertCondition	New + Extraction	Option 1: 1. Add method to allow dynamically created parameters 2. Modify constructor to initialize parameters 3. Modify toXML to serialize parameters 4. Add fromXML method to deserialize a condition	Medium	Low	2	$
6	Query	New + Extraction	Option 2: - Add class for nodes to represent a task - Add class for nodes to represent a task status - Modify xml2Query class to serialize task and task status	Medium	Medium	2	$
7	Alert	New + Extraction	Option 2: - Add triggers to send an alert to alert component - Make changes to constructor to deserialize task and task status	Medium	Medium	2	$
8	AlertEngine	New + Extraction	Option 2: - Send alert to alert component	Medium	Medium	2	$
			TOTALS				
			Option 1 for SetTaskAlert			**20**	
			Option 2 for SetTaskAlert			**24**	
			Without SetTaskAlert			**11**	
			Without SetTaskAlert and without separation from PS			**7**	

Develop Migration Strategy


Establish Migration Context
Migration Feasible?
No
Yes
Define Candidate Services
Describe Existing Capability
Describe Target SOA Environment
Analyze the Gap
Develop Migration Strategy

Develop a migration strategy that that makes sense for the organization and addresses the identified migration issues, e.g.

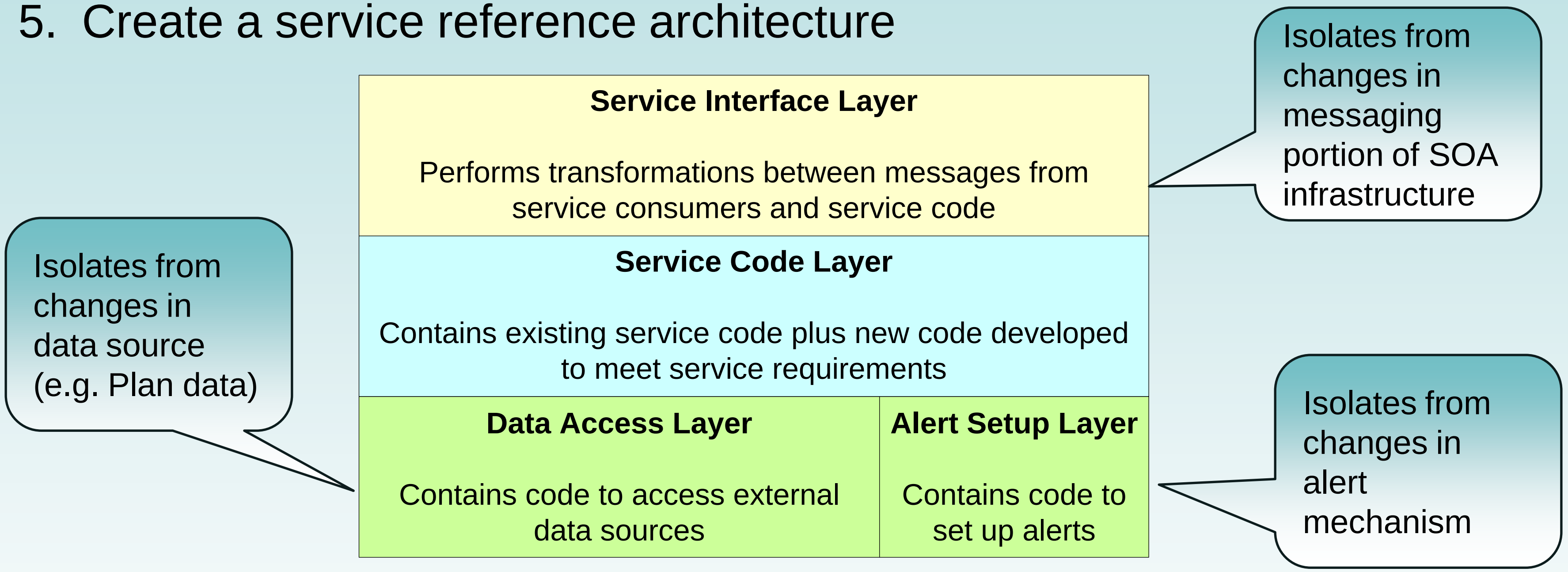
- Feasibility, risk and options for proceeding with the migration effort
- Identification of a pilot project
- Order in which to create additional services
- Guidelines for identification and creation of services
- Options for source of service implementation code
- Mechanisms for providing service functionality
- Specific migration paths to follow
- Needs for additional information, training, technology evaluation, …

Case Study: Migration Strategy $_1$

1. Define scope of initial migration for short-term feasibility demonstration
   - Decision of what services to implement and whether they would have time to separate MSS from PS
2. Define scope of subsequent iterations
   - Will depend on additional services to be created from MSS as well as progress made in the migration of PS to services
3. Finalize service inputs and outputs
   - Alert condition structure was still undefined
4. Gather information about the publish-subscribe component to be used as the mechanism for alert capability
   - Alert mechanism was a big unknown

Case Study: Migration Strategy 2

5. Create a service reference architecture


Isolates from changes in messaging portion of SOA infrastructure
Service Interface Layer
Performs transformations between messages from service consumers and service code
Isolates from changes in data source (e.g. Plan data)
Service Code Layer
Contains existing service code plus new code developed to meet service requirements
Data Access Layer
Contains code to access external data sources
Alert Setup Layer
Contains code to set up alerts
Isolates from changes in alert mechanism

6. Adjust estimates
7. Create MSS services using the service reference architecture
8. Document lessons learned

Agenda

SOA Basics

SMART (Service Migration and Reuse Technique)

Summary ⇦

Summary

SOA offers significant potential for leveraging investments in legacy systems by providing a modern interface to existing capabilities, as well as exposing legacy functionality to a greater number of users

SMART analyzes the viability of reusing legacy systems in an SOA environment:

- Does it make sense to migrate the legacy system to services?
- What services make sense to develop?
- What legacy system components can be used to implement these services?
- What changes to components are needed to accomplish the migration?
- What migration strategies are most appropriate?
- What are the preliminary estimates of cost and risk?
- What is an ideal pilot project that can help address some of these risks?

Resources and Training

SMART Report

- http://www.sei.cmu.edu/publications/documents/08.reports/08tn008.html

Public Courses

- Migration of Legacy Systems to SOA Environments

  http://www.sei.cmu.edu/products/courses/p59b.html

- SMART Training Workshop

  http://www.sei.cmu.edu/products/courses/p73.html

Certification

- SMART Team Lead

  http://www.sei.cmu.edu/certification/soasmart.html

ARCHITECTURE AT ALL SCALES
Fifth SEI Architecture Technology User Network Conference
5th anniversary SATURN
May 4-7, 2009 | Radisson Greentree | Pittsburgh, Pennsylvania

Are you interested in learning more? Visit http://www.sei.cmu.edu/architecture/saturn/ to

SATURN → TECHNOLOGY	Find out about the SEI software architecture work, current research, tools and practices, news, and how the SEI can help you.
SATURN → NETWORK	Stay connected to architecture experts through the SATURN Network on LinkedIn.
SATURN 2009	5th anniversary SATURN	Attend SATURN 2009 – the annual conference that brings together experts from around the world to exchange best practices in developing, acquiring, and maintaining software, systems, and enterprise architecture. Registration is now open!

SOA Topics at SATURN 2009

Course: Migrating Legacy Systems to SOA Environments (Grace Lewis and Dennis Smith, SEI, USA)

Tutorial: Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing (Doug Schmidt, Vanderbilt University, USA)

Papers

- Career Track for Architects in IT Service Provider Organizations (Shankar Kambhampaty, Satyam Computer Services Limited, India)
- How Acquisition Practice Can Impede SOA Governance (Lloyd Brodsky, CSC, USA)

SATURN 2009